Local-in-time Adjoint-based Method for Optimal Control/Design Optimization of Unsteady Compressible Flows

N. K. Yamaleev,*B. Diskin,†and E. J. Nielsen‡

We study local-in-time adjoint-based methods for minimization of flow matching functionals subject to the 2-D unsteady compressible Euler equations. The key idea of the local-in-time method is to construct a very accurate approximation of the global-in-time adjoint equations and the corresponding sensitivity derivative by using only local information available on each time subinterval. In contrast to conventional time-dependent adjoint-based optimization methods which require backward-in-time integration of the adjoint equations over the entire time interval, the local-in-time method solves local adjoint equations sequentially over each time subinterval. Since each subinterval contains relatively few time steps, the storage cost of the local-in-time method is much lower than that of the global adjoint formulation, thus making the time-dependent optimization feasible for practical applications. The paper presents a detailed comparison of the local- and global-in-time adjoint-based methods for minimization of a tracking functional governed by the Euler equations describing the flow around a circular bump. Our numerical results show that the local-in-time method converges to the same optimal solution obtained with the global counterpart, while drastically reducing the memory cost as compared to the global-in-time adjoint formulation.

I. Discrete Optimal Control Problem

We consider time-dependent optimization problems that include both optimal control and shape/design optimization of unsteady compressible flows governed by the 2-D Euler equations. The governing equations are discretized by using a node-centered finite-volume scheme, where solution values are stored at the mesh nodes. Inviscid fluxes at cell interfaces are computed using the upwind scheme of Roe.[1] The discretized Euler equations including the boundary conditions can be written in the following form:

$$\frac{\mathbf{Q}^n - \mathbf{Q}^{n-1}}{\Delta t} + \mathbf{R}^n = \mathbf{0}, \tag{1}$$

where $\mathbf{Q} = \int_V \mathbf{U} dV$, $\mathbf{U}$ is a vector of the conserved variables, V is the control volume, and $\mathbf{R}$ is the spatial undivided flux residual. It should be noted that the above discrete formulation (1) is very general and can be directly applied to the unsteady Reynolds-averaged Navier-Stokes equations. In Eq. (1), the time derivative has been approximated using the implicit first-order backward-difference (BDF-1) formula. Note that 2nd– and 3rd-order BDF formulae can also be used in the present formulation with minor modifications.[2]

The discrete time-dependent optimization problem is formulated as follows:

$$\begin{cases} \min\limits_{\mathbf{D}\in\mathcal{D}_a} F_{\rm obj}(\mathbf{D}), \quad F_{\rm obj}(\mathbf{D}) = \sum\limits_{n=1}^{N} f^n(\mathbf{Q}^n, \mathbf{D})\Delta t \\ \text{s.t. Eq. (1)}, \end{cases} \tag{2}$$

*Associate Professor, North Carolina A&T State University, Member AIAA. E-mail: nkyamale@ncat.edu

†Associate Research Fellow, National Institute of Aerospace, Member AIAA. E-mail: bdiskin@nianet.org; also Visiting Associate Professor, MAE, UVA

‡Research Scientist, NASA Langley Research Center, Senior Member AIAA, E-mail: Eric.J.Nielsen@nasa.gov

where $\mathbf{D}$ is a vector of the control or design variables which may depend on time, $\mathcal{D}_a$ is a set of admissible controls, which depends on specifics of the target physical system and ensures the existence of a solution of the optimization problem, N is the total number of time steps, $\mathbf{Q}$ is the solution of the unsteady, compressible Euler equations (1), F_{obj} is an objective functional. The minimization problem (2) is very general and directly applicable to both active flow control and aerodynamic design optimization of unsteady flows. Similar optimal control and design optimization problems governed by the unsteady incompressible and compressible Euler/Navier-Stokes equations have been considered in.[3–9]

To reduce the complexity of the optimization problem (2), without loss of generality, it is assumed that the objective functional F_{obj} is a scalar quantity. In the present analysis, we consider the following discrete convex functional at each time level:

$$f^n = \sum_{j \in \Gamma_c} \left[C_j^n - \left(C_j^{\text{target}} \right)^n \right]^2 \tag{3}$$

where C_j is an aerodynamic quantity such as lift or pressure on the controlled boundary surface Γ_c, C_j^{target} is a given target value of C_j. In the present paper, only matching objective functionals given by Eq. (3) are considered.

II. Global-in-time Adjoint-based Optimization Method

The discrete time-dependent optimization problem (2) is solved by using the method of Lagrange multipliers which is used to enforce the governing equations and the corresponding boundary conditions (1) as constraints. The discrete Lagrangian functional is defined as follows:

$$L(\mathbf{D}, \mathbf{Q}, \mathbf{\Lambda}) = \sum_{n=1}^{N} f^n \Delta t \quad + \sum_{n=2}^{N} \left[\mathbf{\Lambda}^n \right]^T \left(\tfrac{\mathbf{Q}^n - \mathbf{Q}^{n-1}}{\Delta t} + \mathbf{R}^n \right) \Delta t + \left[\mathbf{\Lambda}^1 \right]^T \left(\mathbf{Q}^1 - \mathbf{Q}^{\text{in}} \right), \tag{4}$$

where $\mathbf{\Lambda}$ is a vector of Lagrange multipliers or costate variables, $n = 1$ corresponds to the initial moment of time, $\mathbf{Q}^{\text{in}}$ is the initial condition for the Euler equations, f^n is the objective functional given by Eq. (3), and $\mathbf{R}^n = \mathbf{R}(\mathbf{Q}^n, \mathbf{D})$ is the spatial undivided residual. Note that the scalar product of the costate vector and the flux residual vector in Eq. (4) can be interpreted as the integral over the computational domain.

The sensitivity derivative is obtained by differentiating the Lagrangian with respect to $\mathbf{D}$, which yields

$$\begin{aligned} \tfrac{dL}{d\mathbf{D}} &= \sum_{n=1}^{N} \left(\tfrac{\partial f^n}{\partial \mathbf{D}} + \left[\tfrac{\partial \mathbf{Q}^n}{\partial \mathbf{D}} \right]^T \tfrac{\partial f^n}{\partial \mathbf{Q}^n} \right) \Delta t + \left[\tfrac{\partial \mathbf{Q}^N}{\partial \mathbf{D}} \right]^T \left(\tfrac{\mathbf{\Lambda}^N}{\Delta t} + \left[\tfrac{\partial \mathbf{R}^N}{\partial \mathbf{Q}^N} \right]^T \mathbf{\Lambda}^N \right) \Delta t \\ &+ \sum_{n=2}^{N-1} \left[\tfrac{\partial \mathbf{Q}^n}{\partial \mathbf{D}} \right]^T \left(\tfrac{\mathbf{\Lambda}^n - \mathbf{\Lambda}^{n+1}}{\Delta t} + \left[\tfrac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \mathbf{\Lambda}^n \right) \Delta t \\ &- \left[\tfrac{\partial \mathbf{Q}^1}{\partial \mathbf{D}} \right]^T \Lambda_2 + \left(\left[\tfrac{\partial \mathbf{Q}^1}{\partial \mathbf{D}} \right]^T - \left[\tfrac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}} \right]^T \right) \mathbf{\Lambda}^1 + \sum_{n=2}^{N} \left[\tfrac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^n \Delta t \end{aligned} \tag{5}$$

Regrouping the terms, Eq. (5) can be recast as follows:

$$\begin{aligned} \tfrac{dL}{d\mathbf{D}} &= \sum_{n=1}^{N} \tfrac{\partial f^n}{\partial \mathbf{D}} \Delta t + \left[\tfrac{\partial \mathbf{Q}^N}{\partial \mathbf{D}} \right]^T \left(\tfrac{\mathbf{\Lambda}^N}{\Delta t} + \left[\tfrac{\partial \mathbf{R}^N}{\partial \mathbf{Q}^N} \right]^T \mathbf{\Lambda}^N + \tfrac{\partial f^N}{\partial \mathbf{Q}^N} \right) \Delta t \\ &+ \sum_{n=2}^{N-1} \left[\tfrac{\partial \mathbf{Q}^n}{\partial \mathbf{D}} \right]^T \left(\tfrac{\mathbf{\Lambda}^n - \mathbf{\Lambda}^{n+1}}{\Delta t} + \left[\tfrac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \mathbf{\Lambda}^n + \tfrac{\partial f^n}{\partial \mathbf{Q}^n} \right) \Delta t \\ &+ \left[\tfrac{\partial \mathbf{Q}^1}{\partial \mathbf{D}} \right]^T \left(\tfrac{\mathbf{\Lambda}^1 - \mathbf{\Lambda}^2}{\Delta t} + \tfrac{\partial f^1}{\partial \mathbf{Q}^1} \right) \Delta t - \left[\tfrac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^1 + \sum_{n=2}^{N} \left[\tfrac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^n \Delta t \end{aligned} \tag{6}$$

For aerodynamic design optimization and optimal control problems, the number of design or control variables may be very large. Therefore, the computation of $\partial \mathbf{Q} / \partial \mathbf{D}$ is extremely expensive in terms of the CPU time, because it requires as many solves of the primary problem as the total number of the design/control variables involved. To eliminate this term from the Lagrangian, the second, third, and forth terms on the right hand side are set equal to zero, thus leading to the following adjoint equations for determining the Lagrange multipliers:

$$\begin{cases} \frac{1}{\Delta t} \mathbf{\Lambda}^N + \left[\frac{\partial \mathbf{R}^N}{\partial \mathbf{Q}^N} \right]^T \mathbf{\Lambda}^N = - \frac{\partial f^N}{\partial \mathbf{Q}^N}, & \text{for } n = N \\ \frac{1}{\Delta t} \left(\mathbf{\Lambda}^n - \mathbf{\Lambda}^{n+1} \right) + \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}^n} \right]^T \mathbf{\Lambda}^n = - \frac{\partial f^n}{\partial \mathbf{Q}^n}, & \text{for } 2 \le n \le N-1 \\ \frac{1}{\Delta t} \left(\mathbf{\Lambda}^1 - \mathbf{\Lambda}^2 \right) = - \frac{\partial f^1}{\partial \mathbf{Q}^1}, & \text{for } n = 1 \end{cases} \tag{7}$$

The main advantage of the adjoint formulation is that at each optimization iteration, the adjoint equations (7) do not depend on $\mathbf{D}$ and should be solved once regardless of the number of the control/design variables. Equations (7) represent a linear system of equations for the adjoint variables, which are solved backward in time. Using the Lagrange multipliers found from Eqs. (7), the sensitivity derivative is calculated as follows:

$$\frac{dL}{d\mathbf{D}} = \sum_{n=1}^{N} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \sum_{n=2}^{N} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}} \right]^T \mathbf{\Lambda}^n \Delta t - \left(\frac{\partial \mathbf{Q}^{\text{in}}}{\partial \mathbf{D}} \right)^T \mathbf{\Lambda}^1, \tag{8}$$

where $\frac{\partial f^n}{\partial \mathbf{D}}$ and $\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}}$ are calculated by using $\mathbf{Q}^n$ stored during the forward sweep in time.

A minimum of the functional is found by using the steepest descent method in which each step of the optimization cycle is taken in the negative gradient direction

$$D^l_{k+1} = D^l_k - \delta_k \left(\frac{dL}{dD^l} \right), \tag{9}$$

where δ_k is an optimization step size which can be chosen adaptively, k is the number of optimization cycle, D^l is an l-th component of the vector the control/design variables, $\mathbf{D}$. Note that for optimal control problems considered herein, D^l denotes the value of the control variable at the l-th time step, so that the dimensionality of the vector $\mathbf{D}$ is equal to the total number of time steps used for integration of the flow equations. The sensitivity derivative dL/dD_l in Eq. (9) is determined using Eq. (8) which requires the solution of the adjoint equations (7). During the solution of the adjoint equations which are integrated backward in time, the sensitivity derivative at each time step is computed and added to its value at the previous time step. At $n = 1$, the complete sensitivity derivative vector is available and used in Eq. (9) for calculating a new value of each control variable D^l_{k+1}. Then, the entire optimization cycle is repeated until $|D^l_{k+1} - D^l_k| < \epsilon$ for all l, where ϵ is a given tolerance. The above procedure can be summarized in the form of the following global-in-time adjoint-based algorithm:

ALGORITHM 1.

1. Choose δ, $\mathbf{D}_1$; set $k = 1$ and $\frac{dL}{d\mathbf{D}} = 0$.
2. Solve Eq. (1) for $\mathbf{Q}^n_k$ for $n = \overline{1, N}$ and store $\mathbf{Q}^1_k, \ldots, \mathbf{Q}^N_k$.
3. Solve Eq. (7) backward in time for $\mathbf{\Lambda}^n_k$, $n = \overline{1, N}$.
4. Evaluate $\frac{dL}{d\mathbf{D}}$ using Eq. (8).
5. Calculate D^l_{k+1} using Eq. (9) for all l.
6. If $\|\mathbf{D}^l_{k+1} - \mathbf{D}^l_k\| > \epsilon$, set $k = k + 1$ and go to (2); otherwise stop.

III. Local-in-time Adjoint-based Optimization Method

As has been mentioned in the foregoing section, at each iteration of the global-in-time adjoint-based optimization method, the flow equations are integrated forward in time while the adjoint equations are integrated backward in time over the entire time interval considered. Since the adjoint equations (7) depend on $\mathbf{Q}^n$, the solution of the flow problem has to be stored for all time levels over which the optimization problem is solved. For realistic 3-D large-time optimal control problems, these storage requirements can quickly become prohibitive. This motivates us to use suboptimal strategies to reduce the memory cost of the global-in-time adjoint-based optimization procedure presented in the foregoing section.

We begin by dividing the entire time interval into M subintervals such that $0 = T_1 < \ldots < T_{M+1} = N\Delta t = T_{\text{final}}$, where $T_m = \Delta t N_m$, $M \leq N$, and Δt is the time step used for integrating the flow and adjoint equations. In general, this partitioning can be chosen so that each subinterval contains one or several time steps of the time-marching scheme used for solving the flow equations. The main idea of the proposed suboptimal strategy is based on the observation that the sensitivity derivative of the the global-in-time Lagrangian can be represented as a sum of the sensitivity derivatives of local-in-time functionals:

$$\frac{dL}{d\mathbf{D}} = \sum_{m=1}^{M} \frac{dL^m}{d\mathbf{D}}, \tag{10}$$

where the local Lagrangian functionals are defined as

$$L^m = \begin{cases} \sum\limits_{n=N_m+1}^{N_{m+1}} f^n \Delta t + \sum\limits_{n=N_m+1}^{N_{m+1}} \left[\hat{\mathbf{\Lambda}}^n\right]^T \left(\frac{\hat{\mathbf{Q}}^n - \hat{\mathbf{Q}}^{n-1}}{\Delta t} + \mathbf{R}^n\right) \Delta t, & \text{for } 2 \le m \le M \\ \sum\limits_{n=1}^{N_2} f^n \Delta t + \sum\limits_{n=2}^{N_2} \left[\hat{\mathbf{\Lambda}}^n\right]^T \left(\frac{\hat{\mathbf{Q}}^n - \hat{\mathbf{Q}}^{n-1}}{\Delta t} + \mathbf{R}^n\right) \Delta t + \left[\hat{\mathbf{\Lambda}}^1\right]^T \left(\hat{\mathbf{Q}}^1 - \hat{\mathbf{Q}}^{\text{in}}\right), & \text{for } m = 1, \end{cases} \tag{11}$$

$\hat{\mathbf{Q}}$ and $\hat{\mathbf{\Lambda}}$ are local conserved and adjoint variables defined on the subinterval $(T_m, T_{m+1}]$, accordingly.

The local-in-time sensitivity derivatives $dL^m/d\mathbf{D}$, $m = \overline{1, M}$ can be evaluated by using an adjoint-based approach similar to one described in the foregoing section. Differentiating each local Lagrangian, L^m, with respect to $\mathbf{D}$, collecting the $\partial\hat{\mathbf{Q}}^n/\partial\mathbf{D}$ terms and setting their coefficients equal to zero yield the following local flow adjoint equations for the subinterval $(T_m, T_{m+1}]$:

$$\begin{cases} \frac{1}{\Delta t}\left(\hat{\mathbf{\Lambda}}^{N_{m+1}} - \tilde{\mathbf{\Lambda}}\right) + \left[\frac{\partial \mathbf{R}^{N_{m+1}}}{\partial \hat{\mathbf{Q}}^{N_{m+1}}}\right]^T \hat{\mathbf{\Lambda}}^{N_{m+1}} = -\frac{\partial f^{N_{m+1}}}{\partial \hat{\mathbf{Q}}^{N_{m+1}}} & \text{for } n = N_{m+1} \\ \frac{1}{\Delta t}\left(\hat{\mathbf{\Lambda}}^n - \hat{\mathbf{\Lambda}}^{n+1}\right) + \left[\frac{\partial \mathbf{R}^n}{\partial \hat{\mathbf{Q}}^n}\right]^T \hat{\mathbf{\Lambda}}^n = -\frac{\partial f^n}{\partial \hat{\mathbf{Q}}^n}, & \text{for } N_m + 1 \le n \le N_{m+1} - 1 \\ \frac{1}{\Delta t}\left(\hat{\mathbf{\Lambda}}^1 - \hat{\mathbf{\Lambda}}^2\right) = -\frac{\partial f^1}{\partial \hat{\mathbf{Q}}^1}, & \text{for } n = 1 \end{cases} \tag{12}$$

Note that the last equation in Eq. (12) is used only on the first subinterval $[T_1, T_2]$ corresponding to $m = 1$. Furthermore, if the local adjoint equations (12) are derived solely based on the local-in-time Lagrangian L^m itself, i.e., no information about the global-in-time Lagrangian is invoked, then $\tilde{\mathbf{\Lambda}}$ in Eq. (12) is identically equal to zero. With the local adjoint variables $\hat{\mathbf{\Lambda}}$ satisfying Eq. (12), the local sensitivity derivative on the subinterval $(T_m, T_{m+1}]$ is calculated as follows:

$$\frac{dL^m}{d\mathbf{D}} = \begin{cases} \sum\limits_{n=N_m+1}^{N_{m+1}} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \sum\limits_{n=N_m+1}^{N_{m+1}} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}}\right]^T \hat{\mathbf{\Lambda}}^n \Delta t, & \text{for } 2 \le m \le M \\ \sum\limits_{n=1}^{N_2} \frac{\partial f^n}{\partial \mathbf{D}} \Delta t + \sum\limits_{n=2}^{N_2} \left[\frac{\partial \mathbf{R}^n}{\partial \mathbf{D}}\right]^T \hat{\mathbf{\Lambda}}^n \Delta t - \left(\frac{\partial \hat{\mathbf{Q}}^{\text{in}}}{\partial \mathbf{D}}\right)^T \hat{\mathbf{\Lambda}}^1, & \text{for } m = 1 \end{cases} \tag{13}$$

The local sensitivity derivative at the m-th time subinterval is calculated using Eq. (13) and added to the sum of the sensitivity derivatives obtained at the previous $(m-1)$ subintervals in accordance with Eq. (10). Once the global sensitivity derivative is available at the last M-th time subinterval, the vector of control/design variables is updated by using the steepest descent method as follows:

$$\mathbf{D}_{k+1} = \mathbf{D}_k - \delta_k \left(\frac{dL}{d\mathbf{D}}\right)_k, \tag{14}$$

where k is the number of an optimization iteration, and δ_k is the optimization step size which is chosen adaptively at each optimization cycle. The steepest descent iterations are repeated until $\|\mathbf{D}_{k+1} - \mathbf{D}_k\| < \epsilon$, where ϵ is a user-specified tolerance.

Comparing Eq. (7) with Eq. (12) and Eq. (8) with Eqs. (10, 13), the following important conclusions can be drawn. First of all, each local-in-time set of the adjoint equations (12) defined on a given time subinterval is independent of the other adjoint equations defined on the rest of the time interval $[0, T_{\text{final}}]$, thus indicating that the local adjoint equations (12) can be solved sequentially starting from $m = 1$ and marching *forward* one subinterval by another up to $m = M$. It should be emphasized that within each subinterval $(T_m, T_{m+1}]$, the local adjoint equations (12) have to be integrated backward in time. The local sensitivity derivatives calculated on each subinterval using Eq. (13) are then summed up to give the sensitivity derivative for the entire time interval $[0, T_{\text{final}}]$. Note that the flow adjoint variables obtained with the local adjoint equations (12) for $m = \overline{1, M}$ and the corresponding total sensitivity derivative given by Eq. (10) are not equal to those of the global-in-time formulation Eqs. (7, 8), i.e., $\hat{\mathbf{\Lambda}} \neq \mathbf{\Lambda}$ on $(T_m, T_{m+1}]$, where $\mathbf{\Lambda}$ denotes the solution of the global adjoint equations (7). Though the above approach only approximates the original global sensitivity derivative given by Eqs. (7, 8), it reduces the memory cost by a factor of M as compared with the global-in-time formulation. Indeed, since the local adjoint equations on each time subinterval $(T_m, T_{m+1}]$ can be solved independently on the adjoint equations defined on the other subintervals, only the flow solutions for the current subinterval, $\hat{\mathbf{Q}} = [Q^{N_m+1}, \ldots, Q^{N_{m+1}}]^T$, have to be stored, thus drastically reducing the memory cost.

The second observation resulted from the comparative analysis of Eqs. (7,8, 12, 13) is that the set of the local adjoint equations (12) for $m = \overline{1, M}$ is identical to the global adjoint equations (7) if $\tilde{\mathbf{\Lambda}}$ in Eq. (12) is set to be $\mathbf{\Lambda}^{N_{m+1}+1}$. In spite of the fact that this approach provides complete consistency of the local and global adjoint equations, it destroys the locality of the adjoint equations (12) defined on each subinterval and therefore requires the same storage as the original global-in-time formulation.

The above considerations suggest that $\tilde{\mathbf{\Lambda}}$ in Eq. (12) should be chosen such that it preserves the locality of the local-in-time adjoint equations on each subinterval $(T_m, T_{m+1}]$ and provides a good approximation of $\mathbf{\Lambda}^{N_{m+1}+1}$. To satisfy these constraints, we have chosen $\tilde{\mathbf{\Lambda}}$ in the following form:

$$\tilde{\mathbf{\Lambda}}_k = \mathbf{\Lambda}_{k-1}^{N_{m+1}+1}, \tag{15}$$

where k is the number of an optimization iteration. In other words, the required vector of adjoint variables at the time level $N_{m+1} + 1$ is taken from the previous iteration of the steepest descent method (14). This choice of $\tilde{\mathbf{\Lambda}}$ significantly reduces the memory cost as compared to the global-in-time adjoint formulation. Indeed, in this case, the flow solution should be stored only at those time levels that belong to the current time subinterval $(T_m, T_{m+1}]$. In addition, M adjoint variables, $\mathbf{\Lambda}_{k-1}^{N_{m+1}+1}$ for $m = \overline{1, M}$, from the previous optimization cycle should also be stored, as follows from Eq. (15). Therefore, the overall memory cost of the proposed methodology is $O(N/M + M)$ versus the $O(N)$ cost of the global-in-time formulation. This estimate suggests that the optimal value for the number of time subintervals is $M = \sqrt{N}$, where N is the total number of time levels. Another advantage of this method is that if the steepest descent method (14) converges, then the solution of the set of the local-in-time adjoint equations approach to the solution of the original global adjoint equations, thus providing full consistency of the local to global formulations.

The above local-in-time strategy for solving the minimization problem (2, 3) can be formulated in the form of the following algorithm:

ALGORITHM 2.

1. Choose δ, $\mathbf{D}_1$, and M; set $m = 1$, $k = 1$, $\mathbf{\Lambda}_0^{N_m+1} = 0$ for $m = \overline{1, M}$, and $\frac{dL}{d\mathbf{D}} = 0$.
2. Solve Eq. (1) for $\mathbf{Q}_k^{N_m+1}, \ldots, \mathbf{Q}_k^{N_{m+1}}$.
3. Solve Eq. (12) with $\tilde{\mathbf{\Lambda}} = \mathbf{\Lambda}_{k-1}^{N_m+1}$ backward in time to find $\mathbf{\Lambda}_k^{N_m+1}, \ldots, \mathbf{\Lambda}_k^{N_{m+1}}$.
4. Store $\mathbf{\Lambda}_k^{N_m+1}$.
5. Evaluate $\frac{dL^m}{d\mathbf{D}}$ using Eq. (13).
6. Set $\frac{dL}{d\mathbf{D}} = \frac{dL}{d\mathbf{D}} + \frac{dL^m}{d\mathbf{D}}$.
7. Set $m = m + 1$, if $m \leq M$ go to step (2); otherwise continue.
8. Calculate $\mathbf{D}_{k+1}$ using Eq. (14).
9. If $\|\mathbf{D}_{k+1} - \mathbf{D}_k\| > \epsilon$ set $m = 1$, $k = k + 1$, $\frac{dL}{d\mathbf{D}} = 0$ and go to (2); otherwise stop.

It should be noted that the above local-in-time adjoint-based algorithm can be directly used for solving both time-dependent optimal control problems whose control variables depend on time and design optimization problems whose design variables are independent of time. This is the key difference between the local-in-time method and the locally optimal scheme and its variants (e.g., see Refs. [3, 4]), which are applicable only to optimal control problems, but cannot be used for design optimization. One of these locally optimal control strategies, which is also known as a receding horizon control technique, is presented next.

IV. Locally Optimal Control Strategy

In this section, the minimization problem (2) is interpreted as an optimal control problem with the following control variables $\mathbf{D} = [D^1, \ldots D^N]^T$, where N is the total number of time steps used for integration of the governing equations (1). Similar to the local-in-time approach presented in the foregoing section, we divide the entire time interval into M subintervals: $0 = T_1 < \ldots < T_{M+1} = N\Delta t = T_{\text{final}}$. Along with

the PDE-constrained global optimal control problem (2), we consider the following M local-in-time optimal control problems:

$$\begin{cases} \min\limits_{\hat{\mathbf{D}}\in\hat{\mathcal{D}}_a,\ t\in(T_m,T_{m+1}]} \hat{F}^m_{\text{obj}}(\hat{\mathbf{D}}), \quad \hat{F}^m_{\text{obj}}(\hat{\mathbf{D}}) = \sum\limits_{n=\bar{N}_m}^{N_{m+1}} f^n(\hat{\mathbf{Q}}^n,\hat{\mathbf{D}})\Delta t \\ \text{s.t. Eq. (1) on } (T_m,T_{m+1}] \ \text{ with the initial condition } \mathbf{Q}^{\text{in}} = \mathbf{Q}^{N_m}, \end{cases} \tag{16}$$

where $\bar{N}_m = 1$ if $m = 1$ otherwise $\bar{N}_m = N_m + 1$, $T_m = N_m\Delta t$, Δt is the time step of the time-marching scheme used for solving the flow equations, $\hat{\mathbf{D}} = \left(D_{\bar{N}_m},\ldots,D_{N_{m+1}}\right)^T$ is the subset of the control variables that belong to the time interval $(T_m,T_{m+1}]$, $\hat{\mathbf{Q}}$ is the solution of the Euler equations over the subinterval $(T_m,T_{m+1}]$.

The local optimal control problems are solved sequentially over each time subinterval using the solution from the previous subinterval as an initial condition. To solve each local control problem (16), we use an adjoint-based gradient method similar to one described in Section II. The discrete Lagrangian for the local optimization problem, the corresponding local adjoint equations, and the local sensitivity derivative on the interval $(T_m,T_{m+1}]$ are identical to those used for the local-in-time adjoint-based method and given by Eqs. (11), (12), and (13), respectively. The locally optimal algorithm can be formulated as follows:

ALGORITHM 3.

1. choose δ_k and M; set $m = 1$, $k = 1$, and $\frac{dL^m}{d\hat{\mathbf{D}}} = 0$;
2. Choose $D_1^{\bar{N}_m},\ldots,D_1^{N_{m+1}}$.
3. Solve Eq. (1) for $\mathbf{Q}_k^{\bar{N}_m},\ldots,\mathbf{Q}_k^{N_{m+1}}$.
4. Solve Eq. (12) with $\tilde{\mathbf{\Lambda}} = \mathbf{0}$ backward in time to calculate $\mathbf{\Lambda}_k^{\bar{N}_m},\ldots,\mathbf{\Lambda}_k^{N_{m+1}}$.
5. Evaluate $\frac{dL^m}{d\hat{\mathbf{D}}}$ using Eq. (13).
6. Set $D^n_{k+1} = D^n_k - \delta_k\frac{d\hat{L}^m}{dD^n_k}$ for $N_m < n \le N_{m+1}$.
7. If $\exists n : |D^n_{k+1} - D^n_k| > \epsilon$ then set $k = k + 1$ and go to (3); otherwise continue.
8. Set $m = m + 1$, if $m \le M$ go to step (2); otherwise stop.

In spite of some similarities between Algorithms 2 and 3, there is one major difference, namely, how local-in-time and locally optimal methods update the control variables. The local-in-time method uses the local adjoint equations (12) to evaluate the global sensitivity derivative on $[0,T_{\text{final}}]$, which provides a search direction in the entire design space and drives the control variables to a minimum of the global-in-time objective functional (2). In contrast to the local-in-time method, the locally optimal technique sequentially solves each local optimization problem (16), which is equivalent to finding a minimum of the local objective functional in the design subspace of a much smaller dimension. The global solution on the entire time interval $[0,T_{\text{final}}]$ is obtained by patching together all the local optimal control solutions $(\hat{\mathbf{Q}}^m,\hat{\mathbf{D}}^m)$, $m = \overline{1,M}$. In general, the locally optimal solution $\hat{\mathbf{Q}}^m$ may not be equal to the solution of the global-in-time optimization problem $\mathbf{Q}^n$, $n = \overline{N_m,N_{m+1}}$ on the same time subinterval. Therefore, $\hat{\mathbf{Q}}^m$ is suboptimal with respect to the original time-dependent optimization problem. However, for the class of flow matching problems considered in this paper, the the locally and globally (in time) optimal adjoint-based methods converge to the same optimal solution. The main advantage of the locally optimal strategy is its efficiency. Indeed, a local optimization problem on each time subinterval is M times smaller as compared to the original problem, thus drastically reducing the memory cost if M is large. The locally optimal strategy can be used even if each subinterval consists of a single time step; such optimization method is hereafter termed a "one-step" method. In this case, the original time-dependent optimization problem is replaced with a sequence of stationary optimal control problems.

In Ref. [3], it has been proven that the locally optimal control strategy similar to one described above converges to the solution of the global-in-time optimal control problem governed by the incompressible Navier-Stokes equations. Although, similar proof is not currently available for the compressible Euler and Navier-Stokes equations, our numerical results presented in the next section show that the locally optimal control strategy works very well for the tracking functional Eq. (2) at supersonic flow regimes.

V. Numerical Results

We consider simple unsteady compressible inviscid flows in a channel with a bump to evaluate the performance of the local-in-time and locally optimal methods for design optimization and flow control problems. For all test problems considered, the freestream Mach number is given by

$$M(t) = M_0 + \Delta M \cos(\omega t), \tag{17}$$

where M_0 is a mean value of the freestream Mach number, ΔM and ω are an amplitude and frequency of freestream Mach number oscillations. Since the freestream Mach number oscillates in time, the entire flowfield is essentially unsteady.

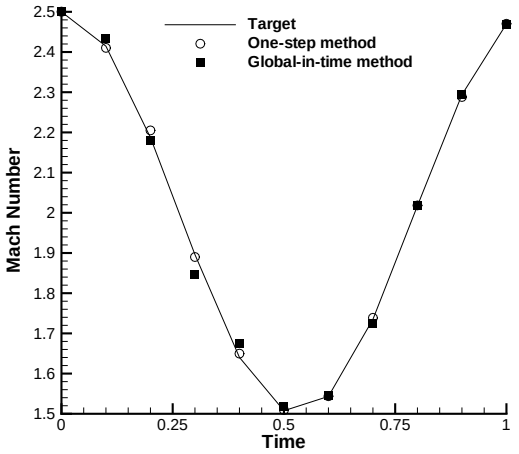

Figure 1. Comparison of the optimal Mach number distributions computed with the one-step and global-in-time methods for the optimal flow control problem.

The test problems are solved on structured grids using a node-centered finite volume code that is first-order accurate both in time and space. At each time step, the nonlinear discrete flow equations are solved by using the Newton's method. For each test, the Euler equations and their adjoints are converged to machine precision. The local adjoint equations are integrated backward in time and require the solution of the Euler equations to be known only for a current time subinterval which is much smaller that the entire time interval. In the present implementation of local optimal and suboptimal methods, only the local unsteady solution set on the current time subinterval is held in operating memory, while for the global-in-time optimization method, the entire unsteady solution history for all time levels has to be stored. The derivatives of $\mathbf{R}$ and f with respect to $\mathbf{Q}$ and $\mathbf{D}$, which are required to form the adjoint equations and the sensitivity derivative, are calculated using the complex variable formula proposed by Lyness.[10]

A. Optimal Flow Control

First, we validate the implementation of the locally optimal method and test its performance for a time-dependent optimal flow control problem given by Eq. (2). For this test case, the thickness of a circular bump is set to be 10% of its chord length, and the final time, T_{final}, is 1. The target solution is computed by using the following parameters in Eq. (17): $M_0 = 2$, $\Delta M = 0.5$, and $\omega = 17\pi/9$. The objective functional is given by

$$f^n = \sum_{j \in \Gamma_c} \left[P_j^n - \left(P_j^{\text{target}} \right)^n \right]^2 , \tag{18}$$

where P_j^n and $\left(P_j^{\text{target}} \right)^n$ are computed and target time-dependent pressure profiles at the lower boundary of the computational domain. Values of the freestream Mach number at each time step $\mathbf{D} = (M_1, \ldots, M_N)^T$ are used as control variables. Both the locally optimal and global-in-time optimization procedures start at a constant freestream Mach number which is set equal to 2.1 for all time levels. The optimization is

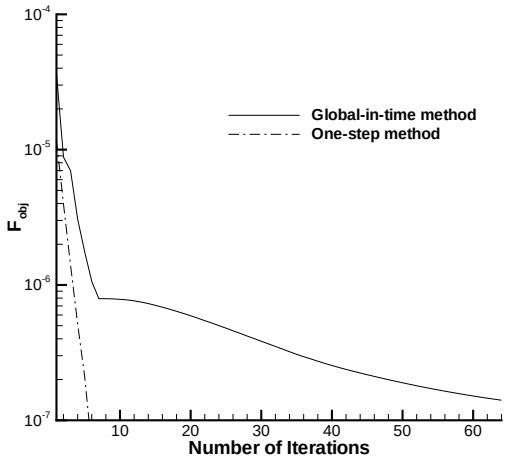

Figure 2. Histories of convergence of the one-step and global-in-time adjoint-based optimization methods.

stopped when either the relative change in the value of each control variable becomes smaller than 10^{-4} or the absolute value of the normalized objective functional $\hat{f}^m(\hat{\mathbf{D}})/(N_{m+1} - N_m)$ becomes smaller than 10^{-7}. In contrast to the global-in-time procedure, the one-step method solves the adjoint equations over one time step, so that only the flow solutions at the current and previous time levels should be stored, thus drastically reducing the memory cost. Note that the computational cost of the global-in-time method per each optimization cycle is N times larger than that of the one-step scheme, where N is the total number of time steps.

To compare the one-step and global-in-time adjoint-based optimization procedures and to evaluate accuracy of each method, the suboptimal and optimal Mach numbers are compared with the target Mach number in Figure 1. As seen in the figure, both methods converge to the target solution on the entire time interval considered, thus validating the one-step and global-in-time adjoint formulations. Note, however, that the one-step strategy provides higher accuracy than its counterpart.

The histories of convergence obtained with both optimization techniques are presented in Fig. 2. For the one-step method, the value of the objective functional drops by an order of magnitude every 3 optimization cycles. Only 6 optimization iterations are needed to reduce the objective functional by two orders of magnitude. The total number of optimization cycles required for convergence of the global-in-time method is an order of magnitude larger than that of the one-step strategy. In spite of the fact that the combined number of optimization iterations of the one-step method for all time levels is approximately the same as the total number of optimization iterations performed by the globally optimal method, each iteration of the one-step method is N times cheaper than that of its counterpart. For $N = 11$ used in this test case, the total CPU time is reduced by about an order of magnitude as compared with the global adjoint-based formulation. This is because at each iteration of the one-step method, the local control problem is solved over one time step, while for the global-in-time method, the flow and adjoint equations are integrated over N time levels. Note that for practical applications that require $10^3 - 10^4$ time steps to resolve the unsteady flow dynamics, the one-step method can provide three to four orders of magnitude reduction in the CPU time and memory cost as compared with its conventional counterpart. To illustrate how the lift coefficient converges to its target value, the optimal C_L obtained with one-step and global-in-time methods as well as the initial and target lift coefficients are presented in Figure 3. The difference between the initial lift coefficient and its target value is of the order of $O(1)$, while the solutions obtained with the one-step and global-in-time methods are almost indistinguishable from the target C_L over the entire time interval considered.

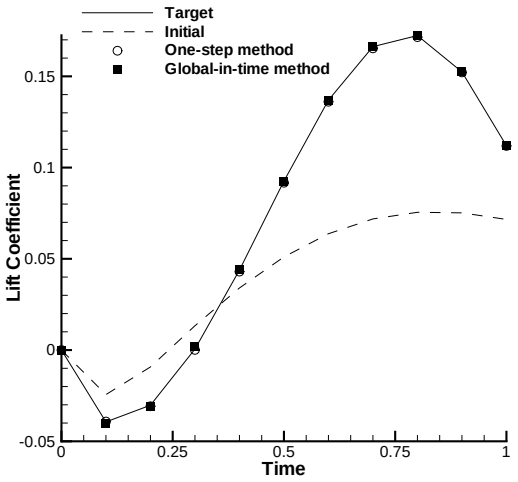

Figure 3. Comparison of the optimal lift coefficients computed using the one-step and global-in-time methods with their initial and target values.

B. Design Optimization

The second test problem has been chosen to evaluate the performance of the local-in-time method for a shape optimization problem given by Eq. (2). The bump shape is described by the following equation:

$$y = d_1\psi_1(x) + d_2\psi_2(x) + d_3\psi_3(x),$$

where $\psi_i(x)$, $i = \overline{1,3}$ are given polynomials satisfying the requirement that the leading and trailing edges of the bump continuously meet the straight lower wall on either side of the bump. The three coefficients d_1, d_2, and d_3 are design variables, i.e., $\mathbf{D} = [d_1, d_2, d_3]^T$. The objective functional is the same as the one used in the previous test problem. Similar to the previous example, the freestream Mach number varies in time, according to Eq. (17) with $M_0 = 2$, $\Delta M = 0.2$, and $\omega = 17\pi/9$. The target pressure distribution is obtained by solving the unsteady 2-D Euler equations with the bump parameters/design variables chosen to be $d_1 = 0.05$, $d_2 = 0.03$, and $d_3 = 0.01$. The initial value of the design variables are set to be zero., i.e., initially, there is no bump on the lower wall. Note that the target flow and bump shape parameters are feasible, and the objective functional vanishes at the optimum; this information is used to monitor the performance of the local and global-in-time optimization methods. The number of time subintervals used in the local-in-time optimization procedure is set to be equal to 5, i.e., each interval consists of only two time steps. The results presented hereafter have been obtained using the simplified form of the local-in-time method with $\tilde{\mathbf{\Lambda}} = \mathbf{0}$ in Eq. (12).

Histories of convergence of the objective functional obtained with the local- and global-in-time methods are presented in Fig. 4. Overall, both methods demonstrate a similar convergence rate. The value of the objective functional monotonically decreases until it becomes less than the specified tolerance which is set to be 10^{-6}. About 50 design cycles are needed to reduce the objective functional by five orders of magnitude. It should be noted that the local-in-time method demonstrates much faster convergence during the first several cycles than its counterpart. Figure 5 shows convergence histories of all three design variables during optimization. The most important conclusion that can be drawn from this comparison is that the local- and global-in-time methods converge to the same solution. From this standpoint, the solution obtained with the local-in-time method is optimal with respect to the original optimization problem (2). It should also be noted that all the design variables approach to their target values. In spite of the fact that the second bump parameter, d_2, is slightly overpredicted while the third one, d_3, is underpredicted, the optimal shape is practically indistinguishable form the target bump, as one can see in Fig. 6. Form the comparisons presented above it follows that the locally optimal method does not only converge to the same optimal solution computed using the global-in-time method, but also reduces the memory cost by a factor 5 as compared with its conventional counterpart.

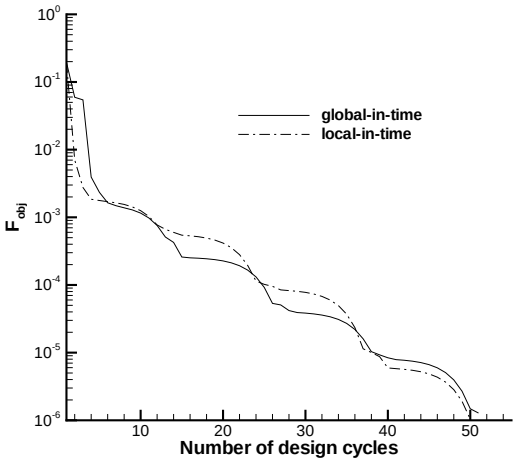

Figure 4. Convergence of the objective functional obtained with the local- and global-in-time methods for the design optimization problem.

References

[1]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Differential Schemes," *J. of Comput. Phys.,* Vol. 43, No. 2, pp. 357-372, 1981.

[2]Nielsen, E. J., Diskin B., and Yamaleev, N. K. "Discrete Adjoint-Based Design Optimization of Unsteady Turbulent Flows on Dynamic Unstructured Grids," submitted to 19th AIAA Computational Fluid Dynamics Conference, San Antonio, TX, 22-25 June, 2009.

[3]Hou, L. S., and Yan, Y., "Dynamics and Approximations of a Velocity Tracking Problem for the Navier-Stokes Flows with Piecewise Distributed Controls," *SIAM J. Control Optim.*, Vol. 35, No. 6, pp. 1847–1885, 1997.

[4]Hinze, M., and Kunisch, K., "On Suboptimal Control Strategies for the Navier-Stokes Equations," in *Control and Partial Differential Equations*, ESAIM, Paris, 1998, pp. 181–198.

[5]Nadarajah, S. K., Jameson, A., "Optimal Control of Unsteady Flows Using a Time Accurate Method," *AIAA Paper* 2002-5436, 2002.

[6]Collis, S. S., Ghayour, K., Heinkenschloss, M., Ulbrich, M., Ulbrich, S., "Optimal Control of Unsteady Compressible Viscous Flows," *Int. J. Numer. Meth. Fluids*, Vol.40, No. 11, 2002, pp. 1401-1429.

[7]Joslin, R.D, Gunzburger, M.D., Nicolaides, R.A., Erlebacher, G., and Hussaini, M.Y., "Self-Contained Automated Methodology for Optimal Control," *AIAA J.* Vol. 35, No. 5, pp. 816–824, 1997.

[8]Mani, K., Mavriplis, D. J., "An Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," *AIAA Paper* 2007-60, 2007.

[9]Rumpfkeil, M.P., and Zingg, D.W., "A General Framework for the Optimal Control of Unsteady Flows with Applications," *AIAA paper*, 2007-1128, 2007.

[10]Lyness, J. N. and Moler, C. B., "Numerical Differentiation of Analytic Functions," *SIAM J. Numerical Analysis*, Vol. 4, pp. 202-210, 1967. 1967, pp. 124–134.

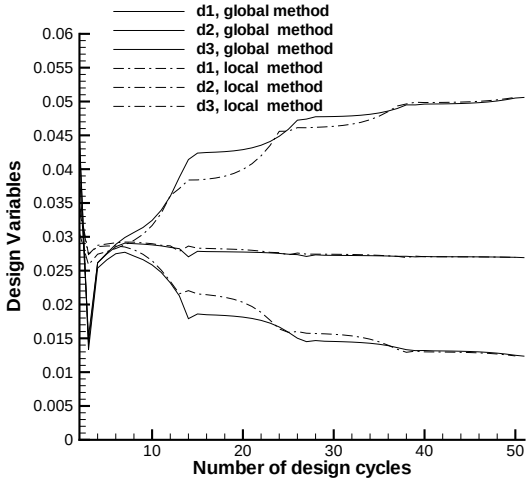

Figure 5. Convergence of the design variables obtained with the local- and global-in-time methods for the design optimization problem.

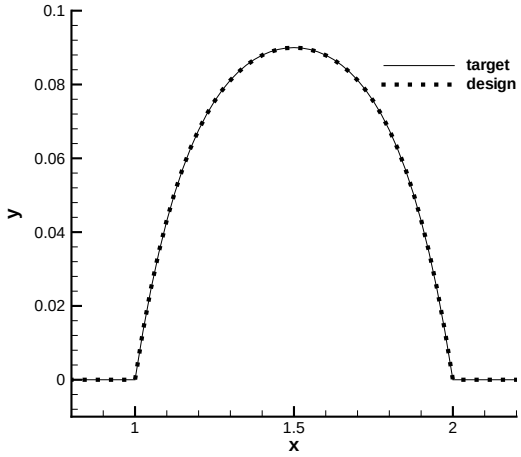

Figure 6. The target and computed bump shapes.